# TECHNO INTERNATIONAL BATANAGAR

## Assignment

Name – Soumyadev Sanyal

Roll – 33200118012

Dept – CSE

Sem – 5th

Year – 3rd

Sub – OS (PCC CS 502)

1) What do you mean by operating system? What are the important functions of an operating system?

—> An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

The important functions of the operating system are:-

a) Memory management
b) Process management
c) Device management.
d) File management
e) Security.
f) Control over system performance.
g) Job accounting.
h) Error detection aids.
i) Co-ordination between other software and user.

2) What do you mean by Multitasking?

—> Multitasking refers to the term where multiple jobs are executed by the CPU simultaneously by switching between them. This switching occurs so frequently that the users may interact with each ~~other~~ programs while it is running.

Soumyadeep Sanyal

3) What do you mean by multiprogramming?

-> When two or more programs are residing in memory at the same time, then sharing the processor is referred to the ~~multiprg~~ multiprogramming. It assumes a single shared processor. It increases CPU utilization by organizing jobs so that the CPU always has one to execute.

4) What do you mean by SPOOLING?

-> Spooling is an acronym for simultaneous peripheral operations on line. It refers to putting data of various I/O jobs in buffer. This buffer is a special area in memory or hard disk which is accessible to I/O devices. Operating system does the following activities, related to distributed environment.

5) What is Process? What are the state of a process and discribe it with suitable picture.

-> A process is a program in execution. It is defined as an entity which represents the basic units of work to be implemented in the system.

Soumyadeep Bongal

# Process States

As a process executes, it changes state. The state of process is defined as the current activity of the process. They are:

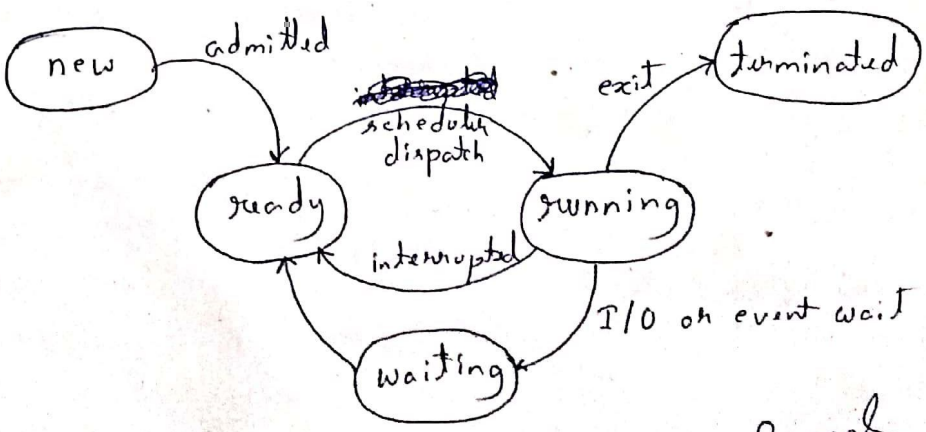
i) New – The process is being created

ii) Ready – The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.

iii) Running – Process instructions are being executed i.e currently being executed.

iv) Waiting – The process is waiting for some event to occur such as the completion of an I/O operation.

v) Terminated – The process has finished execution.




Soumyadeep Sawyal

6) What is process Control Block (PCB). [Describe all information it contain]

→ Each process is represented in the operating system by a process control block, also known as task control block. PCB is the data structure used by operating system. Operating system needs all the informations about particular process.

It contains many pieces of information associated with a specific process which is described below.

a) Pointer - Pointer points to another process control block. It is used for maintaining the scheduling list.

b) Process state - Process state may be new, ready, running waiting and so on.

c) Program Counter - It indicates the address of the next instruction to be executed for this process.

d) CPU registers - CPU registers include general purpose register, stack pointers, index registers and accumulators, etc. Number and type of register totally depends upon the computer architecture.

e) Memory management information - This information may include the value of base and limit registers, the page table, or the segment tables depending on the memory system used by the operating system. This information is useful for de-allocating the memory when the process terminates.

Soumyadeep Bagal.

f> Accounting information – This information includes the amount of CPU and real time used, time limits, job or process numbers, account numbers etc.

7) Write the difference between Long term, short term, and Medium term scheduler.

-> Comparison between scheduler

| Long term scheduler | Short term scheduler | Medium term scheduler |
|---|---|---|
| It is a job scheduler | It is a CPU scheduler | It is a process swapping scheduler |
| Speed is lesser than short term scheduler | Speed is fastest among other two | Speed is in between both short and long term scheduler. |
| It controls the degree of multiprogram. | It provides lesser control ~~income~~ over the degree of multiprogram. | It reduces the degree of multiprogramming. |
| It is almost absent or minimal in the time sharing system. | It is also minimal in time sharing system. | It is a part of time sharing systems |
| It selects processes from pool and loads them into memory for execution | It selects those processes which are ready to execute | It can re-introduce the process into memory and execution can be continued. |

Soumyadeep Sengupta.

8) What do you mean by context switching with proper example?

→ A context switch is the mechanism to store and restore the state or context of a CPU in PCB so that a process execution can be resumed from the same point at a later time. Using this technique a context switcher enable multiple processes to share a single CPU.

Example:

Suppose that multiple processes are stored in a process control block (PCB). One process is running state to execute its task using CPU. As the process is running, another process arrives in the ready queue, which has high priority of completing its task using CPU. Here we use context switching that switches the current process with the new process requiring the CPU to finish its task. While switching the process, a context saves the status of the old process, in register. When the process reloads into the CPU, it starts the execution of the process when the new process ~~holds the~~ stops the old process. If we do not save the state of the process, we have to start its execution at the initial level. In this way, context switching helps the operating system to switch between the processes, store or reloads the process when it requires executing its task.

Soumyadev Sanyal

9) What is thread? Write the difference between process and threads? What are the benefits of the thread.

-> A thread is a flow of execution through the process code, with its own program counter, system registers and stack. It is also called light weight process. It provides a way to ~~improve~~ improve application performance through parallelism. Threads represent a software approach to improving performance of operating system by reducing the overhead thread is equivalent to a classical process.

Difference between process and thread.

| Process | Thread |
|---|---|
| 1) It is heavy weight and resource intensive | 1) It is light weight taking lesser resource ~~from~~ than a process. |
| 2) Process switching needs interaction with operating system. | 2) Thread switching does not need to interact with operating system. |
| 3) In multiple processing environments each process executes the same code but has its own memory and file resources. | 3) All threads can share same set of open files, child processes. |

Soumyadeep Sanyal

4) If one process is blocked then no other process can execute until the first process ~~is~~ is unblocked.

4) While one thread is blocked and waiting, second thread in the same task can run.

5) Multiple processes without using threads use more resources.

5) Multiple threaded processes use fewer resources.

6) In multiple processes each processes operates independently of the others.

6) One thread can read, write or change another threads data.

## Benefits of thread

- Threads minimize context switch time.
- Use of threads provides concurrency within a process.
- Efficient communication.
- More echonomical to create context switch threads.
- Utilization of multiprocessor architectures to a great scale and efficiency.

10) What is user level threads and kernel level threads? Write the difference between them.

-> ~~In this case~~ In the case of user level thread management kernel is not aware of the existence of threads. The thread library contains code for creating and destroying threads for passing message

Soumyadeep Royal

and data between threads, for scheduling thread execution and for saving and restoring thread context. The application begins with a single thread and begins running in that thread.

In the case of Kernel thread the management is done by the kernel. There is no thread management code in the application area. They are directly supported by operating system.

Difference between User level and kernel level

| User level | Kernel level |
|---|---|
| 1) User level threads are faster to create and manage. | 1) Kernel level threads are slower to create and manage. |
| 2) Implementation is by a thread library at the user level. | 2) Operating system supports creation of kernel threads. |
| 3) It is generic and can run on any operating system. | 3) It is specific to the operating system |
| 4) Multi-threaded application cannot take advantage of multi processing | 4) Kernel routines themselves can be multithreaded. |

Soumyadeep Sanyal

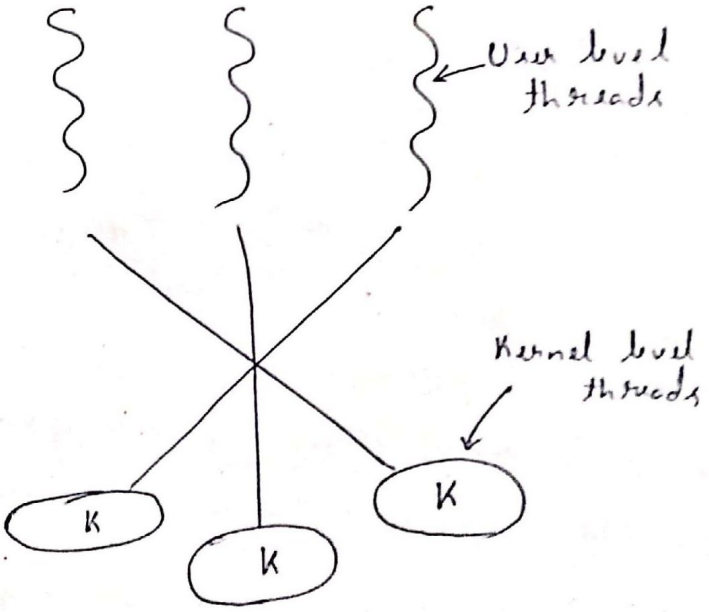
11) What are the relationships in multi-threading models. With proper diagram.

-> Multithreading models are three types:

- Many to many relationship.
- Many to one relationship.
- One to one relationship.

## Many to many model

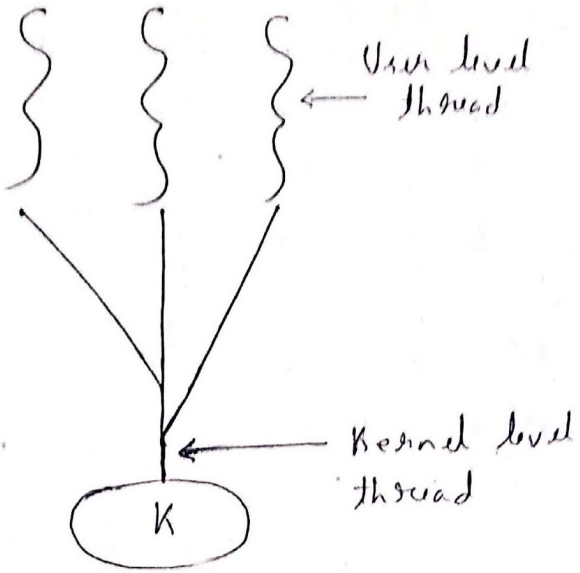
In this model, many user level threads multiplexes to the kernel thread of smaller or equal numbers. The number of kernel threads may be specific to either a particular application or a machine.




## Many to One model

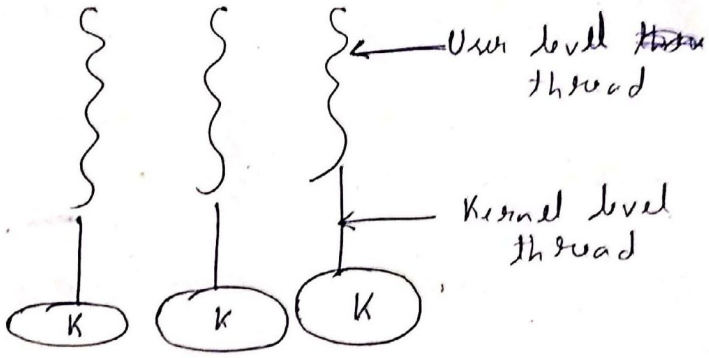
Many to one model maps many user level threads to one kernel level thread. Thread management is done in user space. Only one thread can access the kernel at a time, so multiple threads are unable to run in parallel on multiprocessors.

Soumyadeep Sengupta




One to One Model

There is one to one relationship of user to kernel level thread. It provides more concurrency than many to one relationship.




15) What is semaphore? What are the operations on it? Determine the solution of 'Reader-writers Problem' using semaphore.

→ A semaphore is an integer variable that is used to solve the critical section problem by using two atomic operations

wait()

signal()

Soumyadeep Sanyal

| wait | signal |
| --- | --- |
| This operation is used to acquire. | This operation is used to release the lock. |
| wait (s) {<br>while (s<=0)<br>s--<br>} | Signal (s) {<br>++s;<br>} |

## Solution of reader writer problem

shared data structure:

```
Semaphore rw_mutex = 1;
Semaphore mutex = 1;
int read_count = 0;
```

Write code:

```
while (true) {
    wait (rw_mutex);
    /* writing is performed */
    signal (rw_mutex);
}
```

Reader code:

```
while (true) {
    wait (mutex);
    read_count ++;
    if (read_count == 1)
        wait (rw_mutex);
    signal (mutex);
    /* reading is performed */
    wait (mutex);
    read_count --;
    if (read_count == 0)
        signal (rw_mutex);
    signal (mutex);
}
```

Rowmyadur Rayal

16) Determine the solution of "Dining philosophers problem" using semaphore.

→

```
Semaphore chopstick [5];
while (true) {
    wait (chopstick [i]);
    wait (chopstick [(i+1) % 5]);
/* eat for a while */
    signal (chopstick [i]);
    signal (chopstick [(i+1) % 5]);
```

To avoid deadlock, do either one of the following:

a) Allow at most 4 philosopher.
b) Allow a particular to pick up both or none:
c) Use asymmetric solution odd numbered philosophers pick left chopstick and then right and vice versa,

17) Determine producer consumer solution using semaphore.

→

```
Semaphore empty = 1;
semaphore full = 1;
semaphore mutex = 1;

Producer
    while (true) {
        wait (empty);
        wait (mutex);
/* Producer an item to buffer */
        signal (mutex); signal (full); }
```

Soumya Shrivastava

Consumer:

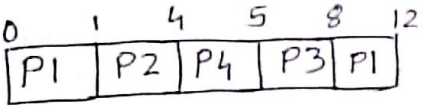
```
while (true) {
    wait (full);
    wait (mutex);
/* consume an item from buffer */
    signal (mutex);
    signal (empty);
}
```

12) Consider the following 4 processes with the length of CPU burst time given in milliseconds. [Preemptive SJF]

| Process | Arrival time | Burst time |
|---|---|---|
| P1 | 0 | 5 |
| P2 | 1 | 3 |
| P3 | 2 | 3 |
| P4 | 4 | 1 |

Find out
1) Average waiting time
2) Average turn around time
3) Average response time.

→ Gantt chart




| Process | AT | BT | CT | TAT | WT | RT |
|---|---|---|---|---|---|---|
| P1 | 0 | 5 | 12 | 12 | 7 | 0 |
| P2 | 1 | 3 | 4 | 3 | 0 | 0 |
| P3 | 2 | 3 | 8 | 6 | 3 | 8 |
| P4 | 4 | 1 | 5 | 1 | 0 | 0 |

Average turn around time = (12+3+6+1)/4
= 5.5

Average waiting time = (7+0+3+0)/4 = 2.5

Avg response time = (0+0+3+0)/4 = .75

Soumyadeep Goyal

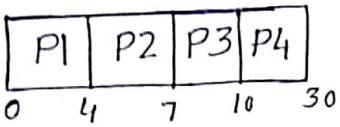
13) Consider the following 3 processes with the length of CPU burst time given in milliseconds.

time quantum [4ms]

| Process | Burst time |
|---|---|
| P1 | 24 |
| P2 | 3 |
| P3 | 3 |

Find Avg waiting time for RR

→ Grantt chart




| Process | BT | CT | WT |
|---|---|---|---|
| P1 | 24 | 30 | 6 |
| P2 | 3 | 7 | 4 |
| P3 | 7 | 10 | 7 |

Avg waiting time = 17/3 = 5.67

14)

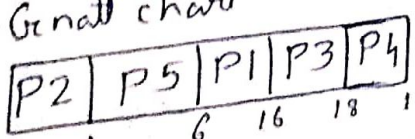
| Process | Burst time | Priority |
|---|---|---|
| P1 | 10 | 3 |
| P2 | 1 | 1 |
| P3 | 2 | 3 |
| P4 | 1 | 4 |
| P5 | 5 | 2 |

Arrival time for all processes is 0

smallest integer = highest priority

Find avg waiting time.

→ Gnatt chart




Avg waiting time = (0+1+6+16+18) = 8.2

Soumyadeep Sanyal